On-Board Preventive Maintenance for Long-Life Deep-Space Missions: A Model-Based Analysis*

Ann T. Tai [†] Leon Alkalai [‡] Savio N. Chau [‡]

[†] IA Tech, Inc.
10501 Kinnard Avenue
Los Angeles, CA 90024

[‡] Jet Propulsion Laboratory
California Institute of Technology
Pasadena, CA 91109

Abstract

The long-life deep-space missions associated with NASA's X2000 Advanced Flight Systems Program creates many unprecedented challenges for us. In particular, the stringent constraints on the mass of a spacecraft and the power on-board preclude traditional fault tolerance approaches which rely on extensive component/subsystem replication, calling for novel, practical approaches to mission reliability enhancement. In this paper, we present an approach to on-board preventive maintenance which rejuvenates a system via periodical duty switching between system components, slowing down a system's aging process and enhancing mission reliability. By exploiting inherent, non-dedicated system redundancy, hardware and software rejuvenation are realized simultaneously without significant performance penalty. Our model-based analysis results confirm a potential for significant gains in mission reliability from on-board preventive maintenance, and provide to us useful insights about the collective effect of age-dependent failure behavior, residual mission life, risk of unsuccessful maintenance and maintenance frequency on mission reliability.

Keywords: On-board preventive maintenance, software/system rejuvenation, analytic modeling, reliability gain, Weibull distribution

*The research described in this paper was carried out by the Jet Propulsion Laboratory, California Institute of Technology, under a contract with the National Aeronautics and Space Administration.

1 Introduction

With NASA's spectacular return to Mars on July 4th 1997, the Mars Pathfinder Lander and its Sojourner Microrover have set a new standard for *Faster, Better, Cheaper* space exploration missions. X2000, a new-generation space technology program aimed at providing an engineering model to multiple long-life deep-space missions [1], is intended to realize the new standard by achieving at least an order of magnitude improvement in both performance and dependability under stringent power and mass constraints. Further, the X2000 technology will enable a high-level efficiency that the cost of the multi-mission purpose spacecraft could be lower than that of the Mars Pathfinder spacecraft. Currently, five missions including Pluto-Kuipar Express, Europa Orbiter, Mars Sample Return, Champollion/DS4, and Solar Probe are becoming the primary users of the X2000 technology. Thus, X2000 is anticipated to supply benchmarks for NASA's new standard.

Particular to the X2000 computing system, reliability is the systems ability to continuously serve a long-life mission with a 8 to 12-year duration and availability is its readiness to serve mission tasks in an unsurveyed deep-space environment such as the planet Pluto and Kuipar disk with an immerse distance from the earth and very limited ground support. In order to simultaneously meet the power, mass and dependability criteria, the X2000 system architecture employs adaptive fault tolerance techniques that exploit non-dedicated system redundancy. To further enhance mission reliability, we have been investigating the notion of *on-board preventive maintenance.* By "on-board preventive maintenance," we refer to the actions taken place during a mission for eliminating or minimizing potential error conditions that accrue over the operational life of a spaceborne system. Although the concept is analogous to that of "software rejuvenation" which has received an appreciable amount of attention for client-server applications in the past few years [2, 3], on-board preventive maintenance concerns both hardware and software, and must be realized in a manner which keeps the maintenance-caused unavailability of a spaceborne system minimal. Specifically, from software perspective, aging phenomenon such as memory leakage and data corruption can be removed via program reinitialization which cleans up the system's internal state [2] (complete age reversal); from hardware perspective, the effects of electronmigration (the driving force of circuit failures), which occurs in microelectronic devices when current density is high, can be reduced through structural/thermal relaxation (self-repair) during a current-off period [4, 5] (partial age reversal). Accordingly, the procedure for realizing on-board preventive maintenance for the X2000 spaceborne computing system involves 1) stopping the running software and host hardware system, and 2) rebooting the system and restarting software execution after a pre-designated period of time. To minimize maintenance-caused system unavailability, we exploit inherent system redundancy. An instance of inherent redundancy is the following: The processor strings designated to jointly perform spacecraft and scientific functions in a non-redundant fashion during a critical mission phase which demands a full computation power (such as the Encountering Phase in Pluto-Kuipar Express during which the spacecraft flies by Pluto), while only a subset of the strings is mandated to be in service during non-critical mission phases such as the cruise phases. Hence, individual processor strings can be scheduled to be on/off duty on a rotation basis for preventive maintenance periodically throughout the mission except during the phase(s) requiring full computation power, virtually having no negative effect on system availability.

In order to study optimal maintenance frequencies that balance the risk of system failure due to component fatigue/aging against that due to unsuccessful maintenance itself, we have conducted model-based analyses. Our preliminary study [6] demonstrated the feasibility of the on-board preventive maintenance approach. For simplicity, the analysis was based on the assumption that the aging processes of both hardware and software component could be completely reversed through preventive maintenance and thus could be treated in the same way in analytic evaluation. Although this assumption suffices the purpose of our preliminary study, we have been investigating into this subject in further depth by discriminating between the aging, age-reversal and failure behavior of hardware and those of software. To accomplish this objective, we utilize Weibull distribution for characterizing both system component's aging and age-reversal processes in a synergistic manner. Further, we derive a recurrence function for mission reliability evaluation which captures the dependencies of system components' aging/failure behavior across duty periods. The recurrence function is simple and can easily lend itself to efficient computer manipulation by utilizing the built-in recursion capability of *Mathematica*$^{\text{TM}}$. To accommodate simultaneous consideration of hardware and software rejuvenation, our model takes into account for the interactions between hardware and software in terms of their failure behavior. Further, we extend our basic model such that the mission profile can be represented in a greater detail. In turn, the resulting analytic framework facilitates the evaluation of the effectiveness of preventive maintenance for a phased mission. Inspired by the results of our earlier study which reveal that optimal duty period is operational environment dependent, we utilize this framework and investigate into the influence of adjusting duty period to mission phases on reliability gain. The evaluation results confirm a potential for significant gains in mission reliability from on-board preventive maintenance, and provide to us useful insights about the collective effect of age-dependent failure behavior, residual mission life, risk of unsuccessful maintenance and maintenance frequency on mission reliability.

The remainder of the paper is organized as follows. Section 2 provides more background information about the X2000 system architecture. Section 3 describes the method for model construction, followed by Section 4 which discusses the results of a model-based evaluation based on the Pluto-Kuipar Express mission profile. The concluding section summarizes what we have accomplished and discusses our plan for future research.

2 Background

One of the major challenges X2000 exhibits to us is the diversity of requirements among the five missions associated with the program which will demand a computation power from a single processor string to three or more, a throughput range from under 20 MIPs to over 100 MIPS, and a mass memory size from 100 Mbytes to 1.5 Gbytes. Therefore, the X2000's computing system architecture must be scalable and distributed in order to accommodate the broad spectrum of requirements. As far as the avionics concern, the X2000 will further advance the packaging technologies initiated by the New Millennium Deep Space One (NMP DS1) program [7, 8]. The NMP DS1 has developed an architecture which consists of a RAD-6000 processor multi-chip-module (MCM), a local memory MCM, a non-volatile mass memory MCM, and an I/O MCM. On the other hand, each X2000 processor string will consist of an processor slice integrated with I/O interfaces, a local memory slice, and one to four non-volatile mass memory. Furthermore, the X2000 architecture has been extended through employing multiple processor strings connected by redundant buses (IEEE 1394 and I2C) to enhance mission reliability [9]. Since the X2000 architecture is scalable through the use of standard buses, it can accommodate from one to more than three processors. An instance of a two-string configuration of the X2000 architecture is depicted in Figure 1.

One of the main feature of the X2000 system architecture is the I/O cross-strapping for the processor strings. Each processor string has dual I/O interfaces, each of which comprises an IEEE 1394 and a I2C interfaces. During a critical mission phase which requires full computation power, all processor strings will be in operation mode to jointly perform the science and engineering functions. On the other hand, during less critical mission phases such as cruise phases, the processor strings will be scheduled on and off duty periodically, serving the mission on a rotation basis. The benefit from doing so are two-fold: 1) a significant saving of the limited power on-board, and 2) periodic rejuvenation of both hardware and software of a processor string.

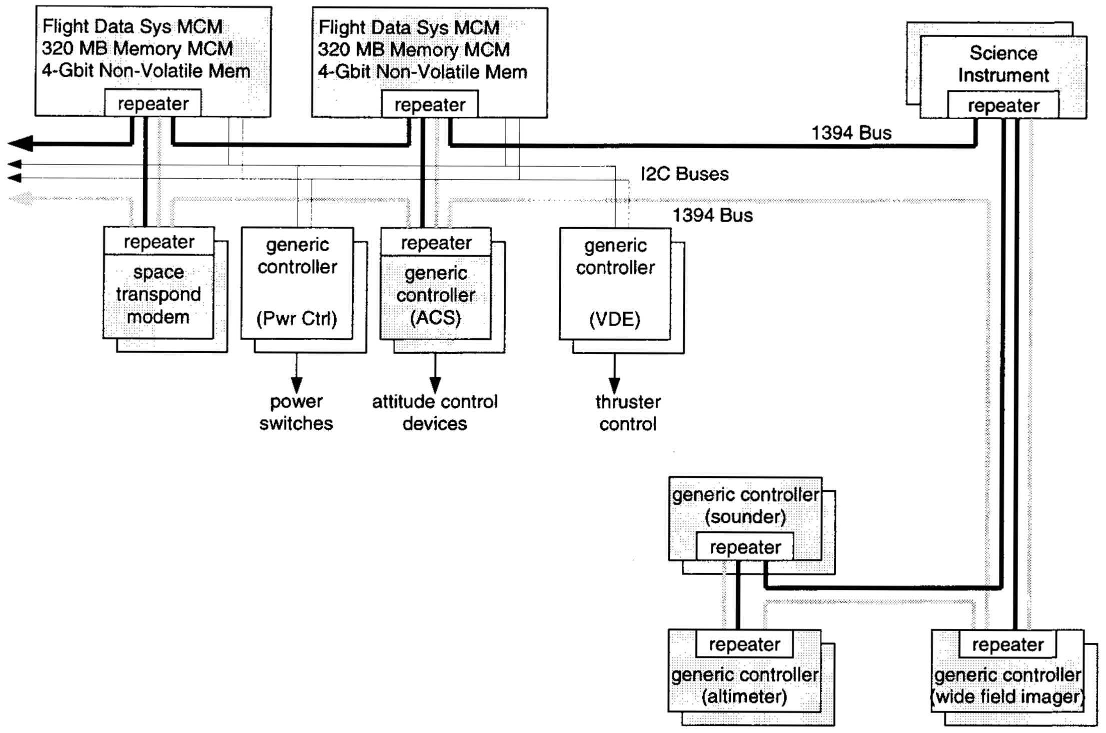

Figure 1: X2000 System Architecture

3 Methods of Model Construction

3.1 Problem Description

The analytic models we develop in this section are based on the system configuration of the X2000 architecture with two processor strings. We first construct a basic model that captures the essence of the on-board maintenance strategies; we then extend the the basic model for a phased mission analysis based on the mission profile of Pluto-Kuipar Express which is to explore the unsurveyed planet Pluto and Kuipar disc (the mission that will travel the longest distance from the earth and have longest duration with respect to the missions associated with the X2000 program).

Due to the combined consideration of time-increasing failure rate and partial age reversal, we yet face a challenge of representing the dependencies between duty periods. Namely, when a new duty period begins, the age of (the hardware part) of the string which just completes its maintenance period and becomes active again is a function of the accumulated length of the duty periods the string has serviced in the past and the amount of age reversal it has obtained from the prior preventive maintenance. Moreover, whether the duty-switching sequence will continue depends upon the availability of resource redundancy (redundancy may become permanently or temporarily unavailable if one of the strings fails or both strings are required to jointly service the mission, respectively). Further, the simultaneous considerations of hardware and software rejuvenation require us to 1) differentiate their aging, age-reversal and failure behavior and, 2) capture the interactions between hardware and software. It is worth to note that we are dealing with a mission having a "phase hierarchy" in the sense that each duty period can be viewed as a mission phase at the lower level while the actual mission phases are defined at the higher level (e.g., the cruise phases and Encountering Phase). Although a number approaches to phased mission analysis were proposed by other researchers (citations here). They are either too restrictive or computationally expensive for our problem. In other words, space applications such as Pluto-Kuipar Express call for analytic methods with greater flexibility and computation efficiency. To accommodate the requirements described above, we propose an approach in which

1. The Weibull distribution is utilized to characterize both aging and age-reversal behavior of a system component in a unified manner.
2. A recurrence function is derived to capture the dependencies between duty periods with respect to the aging, age-reversal and failure behavior of a system component.

Before we proceed to describe the model construction methods in detail in the next two sections, we explain our assumptions as follows.

1) In accordance with the theory that current-off periods will improve a microelectronic device's lifetime [4], we postulate that the amount of age reversal obtained by the hardware of a string through preventive maintenance is directly proportional to the length of a duty period (equivalent to the amount of time a string receives for undergoing maintenance).

2) A string may fail to takeover the workload from the other string during normal duty switching or during failure recovery, causing the system to be in a non-operational state. We call this event "an unsuccessful duty switching" and use the term "switching coverage" to refer to the complement of the probability of such an event.

3) Both hardware and software errors considered in the model are permanent in nature and will cause the corresponding string to be in a non-operational state. Upon the failure of one of the strings, the surviving one will be able to take over the workload. However, if both strings fail, the computing system will be down and lead to an unsuccessful mission.

3.2 Characterizing Aging and Age-Reversal Processes: Utilizing Weibull Distribution

The Weibull distribution is perhaps the most commonly used distribution in reliability engineering because by a proper choice of its shape parameter, an increasing, decreasing or constant failure rate distribution can be obtained [10]. Weibull distribution has been used to describe system behavior with time-increasing failure rate such as fatigue failure, vacuum-tube failure and ball-bearing failure, etc. We further recognize that using Weibull distribution, we are not only able to characterize the strings' age-dependent failure rate by properly setting the shape parameter but also able to represent the age-reversal effect from on-board preventive maintenance by appropriately formulating the "location parameter." Accordingly, we begin model construction with choosing the following form of Weibull probability density function (pdf) [10] that can be exploited to characterize system component's aging and age-reversal processes in a synergistic manner:

$$w(t) = \beta\lambda\left((t-\gamma)\lambda\right)^{\beta-1} e^{-((t-\gamma)\lambda)^{\beta}} \tag{1}$$

where β is the shape parameter (it is set to a value greater than 1 to represent the age-increasing failure rate), λ is the scale parameter and γ is the location parameter that defines the "origin" where the system begins to have the potential of experiencing a failure and to age.

To aid a more precise description, we define the following notation:

T	The *service age* accumulated through its past duty periods.
$\tau[i]$	The *effective age* of a system component (string) at the beginning of duty period $(i+1)$.

By "service age," we mean the actual amount of time during which a system component is on its duty; whereas "effective age" equals to a component's service age minus the amount of "age-reversal" resulting from preventive maintenance. In these terms, the mathematical concepts for aging and partial age reversal processes for a *single string* can be illustrated by Figure 2. In the figure, the abscissa marks the service age of a component while the ordinate measures the Weibull failure rate. The failure rate is both age-dependent and duty-period variant:

$$\lambda_i(S) = \beta\lambda\left((S-\gamma_i)\lambda\right)^{\beta-1}$$

where i is the sequence number for a duty period, S is the component's service age and γ_i is the quantity of partial age reversal (accrued) through the preventive maintenance prior to

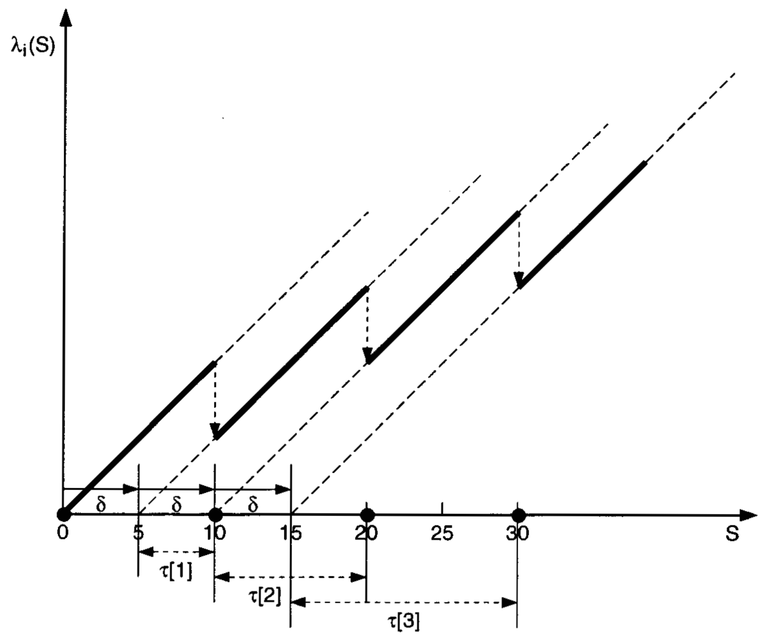

Figure 2: Partial Age Reversal from Service-Age Perspective

duty period i. For clarity of illustration, the shape parameter β is set to 2 such that $\lambda_i(S)$ is linearly increasing within a duty period. The solid dots mark the beginning of each duty period (presuming that a duty period has a duration of 10 weeks). The lines with arrows at the right hand side illustrate the partial age-reversal effect from preventive maintenance. More precisely, these lines describe the following scenario: by the time when a string is ready to start a new duty period i, its age has been reversed by δ time units (5 weeks in this example) as if its "birthday" (corresponding to the value of the location parameter γ_i) moves forward along the service-time horizon due to the effect of the preventive maintenance just completed (such that it becomes "younger"). The thick solid lines represent the effective failure rate (a function of a system's effective age) for the string. Note that they drop down through preventive maintenance that partially reverses the age of a string. The thick dashed lines mark the failure rate corresponding to the case in which preventive maintenance is absent. The dashed lines with arrows at both ends represent the effective age $\tau[i]$ of the string, at the beginning of the $(i+1)th$ duty period.

Figure 3 illustrates the effective Weibull failure rate of a string and the effect of partial age reversal resulting from preventive maintenance in view of mission calendar time (instead of accumulated service time of a single string), where the solid dots mark the beginning of the first string's duty periods and the small circles mark those for the second string.

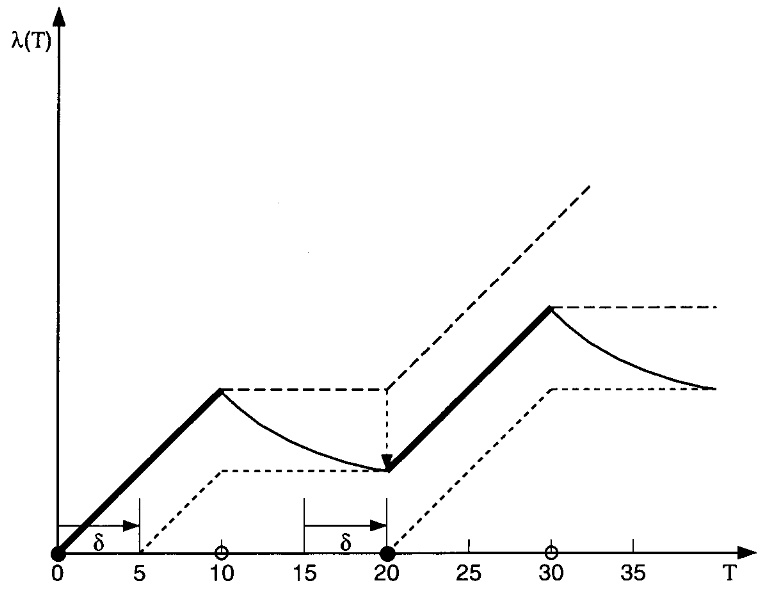

Figure 3: Partial Age Reversal from Calendar-Time Perspective

3.3 Capturing Dependencies between Duty Periods: A Recurrence Function Approach

3.3.1 Basic Model

The series-parallel graph in Figure ?? reveals that the system's behavior with respect to the duty periods is a regenerative renewal process [11]. Accordingly, we can translate the series-parallel graph into a duty-period oriented timing diagram describing the renewal process as shown in Figure 4. The notation used in the illustration are defined below:

θ	The length of mission life.
n	The number of duty periods.
ϕ	The duration of a duty period.
δ	The amount of age reversal resulting from preventive maintenance.
k	The number of successful duty periods (a successful duty period means that a string does not fail when it is on duty and the switching process at the end of the duty period is also successful).
x	The service age of the hardware of a string (which fails first) at the time of failure.
y	The service age of the hardware of a string (which fails second) at the time of failure.

u The service age of the software of a string (which fails first) at the time of failure.

v The service age of the software of a string (which fails second) at the time of failure.

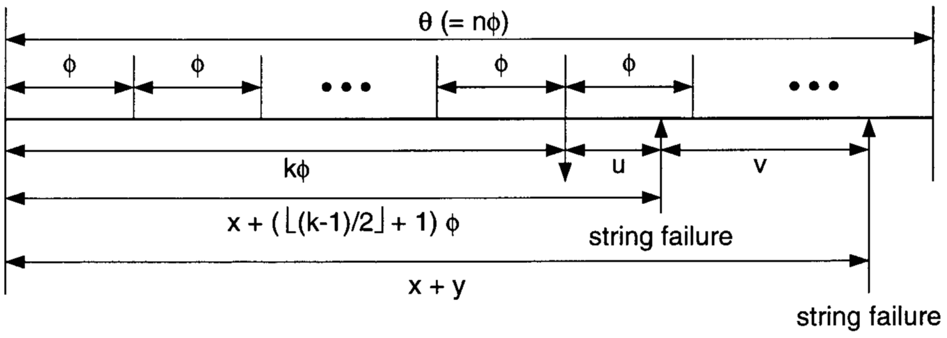

Figure 4: Duty-Period Oriented Timing Diagram

The timing diagram states the success/failure scenarios in terms of duty period and service ages of hardware and software, and illustrates the relationships between them. Namely,

1) $k\phi$ marks the first k duty periods during which both strings are operational.

2) $x + (\lfloor \frac{k-1}{2} \rfloor + 1)\phi$ (or interchangeably, $k\phi + u$) is the time to the first string failure due to a hardware or software error.

3) $x + y$ (or interchangeably, $k\phi + u + v$) is the time to second string failure due to a hardware or software error.

Based on this timing diagram, we can analyze the system's success/failure scenarios as follows.

S1: $k = n \Rightarrow$ The mission succeeds with both strings being operational throughout the mission.

S2: $(k < n) \land (x + y > n\phi) \Rightarrow$ One string fails during the $(k+1)^{th}$ duty period due to a hardware or software error and the other string remains operational through the remainder of the mission.

S3: $(k < n) \land (x + y \leq n\phi) \Rightarrow$ One string fails during the $(k+1)^{th}$ duty period and the other string subsequently fails before the end of the mission.

Let $R(\theta)$ denote the reliability of a mission with a duration ϕ, clearly

$$R(\theta) = P(\text{S1}) + P(\text{S2}). \tag{2}$$

Before we proceed to derive the solution for $R(\theta)$, we introduce the following notation

$T_1[i]$	The service age of a string at the beginning of the i^{th} duty period.
$T_2[i]$	The service age of a string at the end of the i^{th} duty period.

Because a string will be on and off duty periodically,

$$T_1[i] = \left\lfloor \frac{i-1}{2} \right\rfloor \phi, \quad T_2[i] = \left(\left\lfloor \frac{i-1}{2} \right\rfloor + 1 \right) \phi$$

Then we define $F[i]$ as the probability that a string fails due to a hardware error during the ith duty period (of length ϕ), it follows that

$$F[i] = \int_{T_1[i]}^{T_2[i]} w_i(x)dx \tag{3}$$

Note that w_i is the Weibull probability density function (pdf) characterizing hardware's failure behavior in the ith duty period. That is,

$$w_i(t) = \lambda\beta \left(\lambda(t-\gamma_i)\right)^{\beta-1} e^{-(\lambda(t-\gamma_i))^{\beta}} \tag{4}$$

where $\gamma_i = \left\lfloor \frac{i-1}{2} \right\rfloor \delta$ is the amount of age reversal gained by the string, which is in service in the ith duty period, from preventive maintenance by the beginning of that duty period.

Further, let $G[i]$ denote the conditional probability that system fails during duty period i (of length ϕ) given that it remains up by the end of duty period $(i-1)$. $G[i]$ can then be solved in terms of a recurrence function that captures the dependencies between duty periods with respect to strings' aging, age-reversal and failure behavior. More precisely,

$$G[i] = \frac{F[i]}{\prod_{k=1}^{i-1}(1-G[k])}, \quad i \geq 2 \tag{5}$$

with

$$G[1] = F[1].$$

Thus the first term in Equation (2) can be evaluated via a product-form expression

$$P(\mathrm{S1}) = c^{n-1} \prod_{i=1}^{n} (1 - G[i])(1 - Q[i]) \tag{6}$$

where $Q[i]$ is the probability that a string fails due to a software error during the ith duty period. Since software is able to obtain a complete age reversal through rejuvenation, its aging and age-reversal behavior between duty periods are independent, which allows Q to have a simpler expression (compared with $G[i]$), namely,

$$Q[i] = \int_0^{\phi} w_s[t]\, dt = \int_0^{\phi} \mu\alpha(\mu t)^{\alpha-1} e^{(\mu t)^{\alpha}}\, dt, \quad i = 1, 2, \cdots, n \tag{7}$$

To derive the second term in Equation (2) requires us to consider the interaction between hardware and software failure behavior. Accordingly, we define the following terms:

$H[i]$	The probability that a string fails during the i^{th} duty period due to a hardware error given that the software functioning properly up to the time of failure, but the surviving string remains operational throughout the mission.
$S[i]$	The probability that a string fails during the i^{th} duty period due to a software error given that the underlying hardware is operational up to the time of failure, but the surviving string remains operational throughout the mission.

The solutions of $H[i]$ and $S[i]$ are then given by

$$\begin{aligned} H[i] &= \int_{T_1[i]}^{T_2[i]} w_i[x] \left(1 - \int_0^{x-T_1[i]} w_s(t)\, dt\right) \\ &\quad \left(1 - \int_{T_1[i]}^{\theta-x} w_i[y]\, dy\right) \left(1 - \int_0^{\theta-(T_1[i]+x)} w_s(t)\, dt\right) dx \end{aligned} \tag{8}$$

$$\begin{aligned} S[i] &= \int_0^{\phi} w_s[u] \left(1 - \int_{T_1[i]}^{T_1[i]+u} w_i[t]\, dt\right) \\ &\quad \left(1 - \int_0^{\theta-(2\,T_1[i]+u)} w_s(v)\, dv\right) \left(1 - \int_{T_1[i]}^{\theta-(T_1[i]+u)} w_i[z]\, dz\right) du \end{aligned} \tag{9}$$

Note that in the above equations the probabilistic measure of a hardware-fault caused failure is conditioned by the event that the software running on it functions properly up to the time of failure. Likewise, the probabilistic measure of a software-fault caused failure is conditioned by the event that the host hardware remains up by the time of failure. Note also that the upper and lower limits of the integrals are defined in a way such that hardware and software's age-dependent failure behavior can be captured and discriminated from each other. For example, in those terms concerning software success/failure behavior in Equations (8) and (9) (i.e., the integrals of w_s) the lower limits of the integrals are zero while the upper limits correspond to software's service ages starting from duty switching, complying with the assumption that software can obtain a complete age reversal through reinitialization.

To this end, the second term of Equation (2) can be expressed as

$$P(\mathrm{S2}) = \sum_{j=1}^{n} c\,(H[j] + S[j]) \tag{10}$$

In turn, the measure we seek to evaluate, $R(\theta)$, can then be solved analytically.

3.3.2 Phased-Mission Analysis

Now we extend our basic model described in the last section for phased-mission analysis. The Pluto-Kuipar Express consists of three phases, namely, the first cruise phase (Cruise-1), Encountering Phase and second cruise phase (Cruise-2), as shown in Figure 5. While the duration of each of the cruise phases is nearly 6 years, the Encountering Phase has only a very short duration of 4 hours. However, the Encountering Phase is the most critical to the mission since the crucial activities such as orbit maneuver, pointing and s/c control will take place. Therefore, as mentioned in Section 1, both of the strings will be powered on and in full operation throughout this phase.

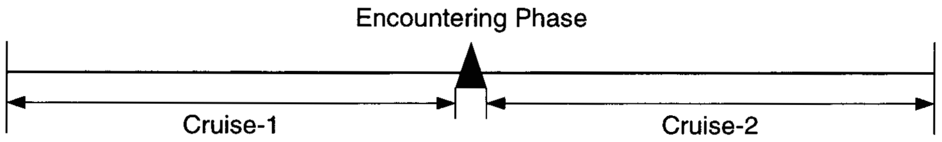

Figure 5: Mission Phases

The additional notation used in describing the phased-mission analysis is defined as follows.

n_1	The number of duty periods in Cruise-1.
n_2	The number of duty periods in Cruise-2.
ϕ_1	The duration of a duty period in Cruise-1.
ϕ_2	The duration of a duty period in Cruise-2.

Φ	The duration of the Encountering Phase. The duration of the *i*th duty period.
δ_1	The amount of age reversal a string obtains from preventive maintenance in Cruise-1.
δ_2	The amount of age reversal a string obtains from preventive maintenance in Cruise-2.

Clearly,

$$\phi_1 = \frac{(\theta - \Phi)/2}{n_1}, \qquad \phi_2 = \frac{(\theta - \Phi)/2}{n_2}$$

And in accordance with the assumption we made for age reversal (see Section 3), we have

$$\delta_1 = \rho\phi_1, \qquad \delta_2 = \rho\phi_2$$

where the coefficient ρ has a domain $[0, 1]$.

Recall that unlike the cruise phases during which normally one string is on duty and the other undergoes maintenance, the Encountering Phase requires both strings to be on duty, implying that there is no preventive maintenance. Therefore, special treatment is required for the Encountering Phase. To preserve the generality of the equations developed for the basic model described in the previous section, we view the Encountering Phase as a special phase which can be "unfold" and thus will become two "parallel duty periods." While the parallel duty periods can be treated as two individual duty periods at the lower level where strings' aging and failure behavior are described, they are considered as a single duty period at the higher level where the solution of mission reliability is derived. It follows that the basic model for solving $R(\theta)$ can be adapted to accommodate phased-mission analysis via the following minor modifications (where the notion of folding and unfolding the two parallel duty periods is reflected).

$$\gamma[i] = \begin{cases} \left\lfloor \frac{i-1}{2} \right\rfloor \delta_1 & i \le n_1 \\ \left\lfloor \frac{i-2}{2} \right\rfloor \delta_1 & i \in \{n_1 + 1, n_1 + 2\} \\ \left\lfloor \frac{n_1}{2} \right\rfloor \delta_1 + \left\lfloor \frac{(i-(n_1+2))-1}{2} \right\rfloor \delta_2 & n_1 \in \{2l \mid l = 1, 2, \cdots\}, \\ & i \in \{n_1 + (2l+1) \mid l = 1, 2, \cdots\} \\ \left\lfloor \frac{n_1-1}{2} \right\rfloor \delta_1 + \left\lfloor \frac{i-(n_1+2)}{2} \right\rfloor \delta_2 & \text{otherwise} \end{cases}$$

$$T_1[i] = \begin{cases} \left\lfloor \frac{i-1}{2} \right\rfloor \phi_1 & i \leq n_1 + 2 \\ \left(\left\lfloor \frac{n_1}{2} \right\rfloor + 1 \right) \phi_1 + \Phi + \left\lfloor \frac{(i-(n_1+2))-1}{2} \right\rfloor \phi_2 & n_1 \in \{2l+1 \mid l = 0,1,2,\cdots\}, \\ & i \in \{n_1 + 2(l+2) \mid l = 0,1,2,\cdots\} \\ \left\lfloor \frac{n_1}{2} \right\rfloor \phi_1 + \Phi + \left\lfloor \frac{(i-(n_1+2))-1}{2} \right\rfloor \phi_2 & \text{otherwise} \end{cases}$$

$$T_2[i] = \begin{cases} \left(\left\lfloor \frac{i-1}{2} \right\rfloor + 1 \right) \phi_1 & i \leq n_1 \\ \left\lfloor \frac{i-1}{2} \right\rfloor \phi_1 + \Phi & i \in \{n_1+1, n_1+2\} \\ \left(\left\lfloor \frac{n_1}{2} \right\rfloor + 1 \right) \phi_1 + \Phi + \left(\left\lfloor \frac{(i-(n_1+2))-1}{2} \right\rfloor + 1 \right) \phi_2 & n_1 \in \{2l+1 \mid l = 0,1,2,\cdots\}, \\ & i \in \{n_1 + 2(l+2) \mid l = 0,1,2,\cdots\} \\ \left\lfloor \frac{n_1}{2} \right\rfloor \phi_1 + \Phi + \left(\left\lfloor \frac{(i-(n_1+2))-1}{2} \right\rfloor + 1 \right) \phi_2 & \text{otherwise} \end{cases}$$

By defining one more boundary condition, Equation (5) can be adapted for the phased-mission analysis as follows.

$$G[i] = \frac{F[i]}{\prod_{k=1}^{i-1}(1 - G[k])}, \quad i \geq 2, i \neq n_1 + 1 \tag{11}$$

with

$$G[n_1+1] = \sum_{j=n_1+1}^{n_1+2} \frac{F[j]}{\prod_{k=1}^{n_1}(1 - G[k])}$$

and

$$G[1] = F[1].$$

Letting $n = n_1 + n_2 + 1$, the measure we seek for can be finally obtained using Equations (2), (6) and (10) derived in Section 3.3.1.

4 Evaluation and Discussion

For a 12-year (624-week) mission, the effectiveness of on-board preventive maintenance is evaluated with respect to the relationships between mission reliability gain and preventive-maintenance frequencies (equivalently, durations of duty periods). First, we study the influence from the strategy of adjusting maintenance frequency to mission phase change on reliability gain. That is, $R(\theta)$ is evaluated along two dimensions — against varying maintenance frequencies for Cruise-1 and Cruise-2 (n_1 and n_2). The value assignment for other parameter are shown in Table 1.

Table 1: Parameter Assignment (I)

θ	β	λ	α	μ	ρ	c
624	5.5	0.00125	5.0	0.00015	0.80	0.99999999

Table 2 (where $R_0(\theta)$ denotes the baseline mission reliability — reliability for a mission without preventive maintenance) and Figure 6 (with n_1 and n_2 plotted in a logarithm scale) illustrated the evaluation results, while Figure 7 provides a more detailed view for the frequencies which lead to the optimal mission reliability. These illustrations reveal that extremely low and extremely high maintenance frequencies will have a detrimental effect on mission reliability. Specifically, when duty switching is carried out only when the mission enters Cruise-2 (corresponding to the case where $n_1 = n_2 = 1$) or when the duration of a duty is 5.3 hours (corresponding to the case where $n_1 = n_2 = 10000$), the resulting mission reliability becomes poorer than the case where preventive maintenance is absent. This can be explained via the tradeoffs between component reliability improvement due to preventive maintenance and the likelihood of system failure caused by an unsuccessful duty switching. More precisely, for the extremely low maintenance frequency case, the component reliability improvement is too insignificant to compensate the risk of unsuccessful duty switching caused mission failure; on the other hand, for the extremely high maintenance frequency case, the excessive mission failure risk associated with unsuccessful duty switching overweighs the benefit from preventive maintenance.

A more interesting observation with regard to adjusting maintenance frequency to mission phase is as follows. Intuitively, a later mission phase favors more frequent maintenance (thus a shorter duty period) due to a higher vulnerability of system failure deriving from component aging. Contrarily, our analyses reveal that it will not be beneficial in general to increase maintenance frequency (i.e., to decrease the duration of a duty period) in the later mission life. Indeed, a strategy that exercises preventive maintenance in a less frequent manner in the later mission life usually leads to an optimal mission reliability. This surprising result stems from some tradeoffs among system attributes which may not be obvious without analytic modeling. Specifically, the likelihood of system failure before the end of a mission tends to 1) increase as the age-dependent failure rate increases and, 2) decrease as the residual mission life decreases. In other words, while the system gets more vulnerable to failure as it is aging, the corresponding decreasing residual mission life allows less frequent maintenance because the resulting reliability improvement becomes relatively less significant, which may not compensate the risk of losing the mission due to unsuccessful duty switching.

Table 2: Adjusting Duty Period to Mission Phase, $R_0(\theta) = 0.9998671833$

n_1,n_2	1	20	50	1000	10000
1	0.9998356063	0.9999170825	0.9999168921	0.9999074993	0.9998180132
20	0.9999925103	0.9999967721	0.9999966254	0.9999872167	0.9998972708
50	0.9999929806	0.9999972368	0.9999970902	0.9999876814	0.9998977353
1000	0.9999842385	0.9999883192	0.9999881721	0.9999787631	0.9998888177
10000	0.9998959872	0.9998984049	0.9998982537	0.9998888433	0.9997989059

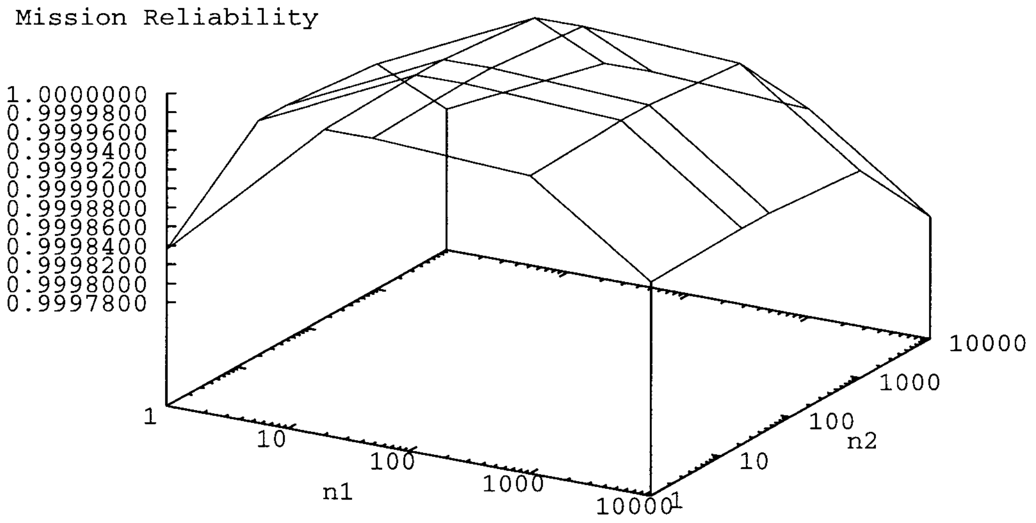

Figure 6: Optimal OBPM Frequency (I)

Next we compare the reliability gain from preventive maintenance and its relationship with phase-adjusted maintenance frequency for two scenarios: in the first and second scenarios, hardware and software faults dominate system failure, respectively. The parameter value assignments are shown in Tables 3 and 4; whereas the evaluation results are displayed in Tables 5 and 6, respectively.

Contrasting Table 5 with Table 6, we note that,

1. Reliability gain for the software-fault dominating case is more significant than the hardware-fault dominating case (2 orders versus 3 orders of magnitude).

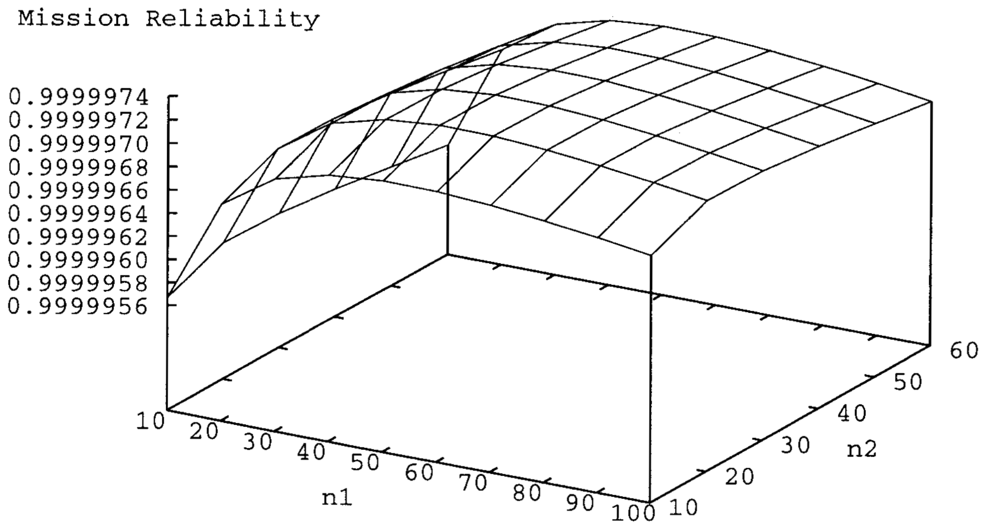

Figure 7: Optimal OBPM Frequency (II)

Table 3: Parameter Assignment (II)

θ	β	λ	α	μ	ρ	c
624	5.0	0.0015	5.0	0.0005	0.45	0.9999999

2. Optimal mission reliability can be achieved with a lower preventive maintenance frequency for the software fault dominating case, relative to the hardware fault dominating case. (For the former case, the optimal maintenance frequencies for Cruise-1 and Cruise-2 are 12 and 6, corresponding to the duty periods of 52 weeks and 104 weeks, respectively; whereas for the latter case, the optimal maintenance frequencies for Cruise-1 and Cruise-2 are 60 and 40, corresponding to the duty periods of 10.4 weeks and 15.6 weeks, respectively.)

3. The software-fault dominating case allows a more significant reduction in preventive maintenance frequency for Cruise-2, relative to Cruise-1 (50% reduction versus 30%

Table 4: Parameter Assignment (III)

θ	β	λ	α	μ	ρ	c
624	5.0	0.0005	5.0	0.0015	0.45	0.9999999

reduction for the software and hardware dominating cases, respectively) for optimal reliability gain.

Explanation of these distinctions resides in the differing characteristics of aging and age-reversal behavior of hardware and software. Specifically,

1. Software can attain, through preventive maintenance, a complete age-reversal that hardware is unable to achieve. As a result, reliability gain from preventive maintenance for software is more significant and can be achieved with less frequent maintenance.

2. By the same token, hardware's vulnerability of failure will be greater in a later mission phase, while software's aging and failure behavior patterns will remain invariant in the sense that its performance can always be completely restored through rejuvenation. Consequently, the collective effect of aging and age-reversal behavior, residual mission life and switching coverage permits a more significant reduction of maintenance frequency in the later mission phase for the software-fault dominating case.

Table 5: Optimal Duty Period for Hardware Fault Dominating Case, $R_0(\theta) = 0.9980540436$

n_1,n_2	1	40	60	1000	10000
1	0.9975612227	0.9987574853	0.9987562969	0.9986657264	0.9977864585
40	0.9998196221	0.9999255154	0.9999248241	0.9998335646	0.9989379078
60	0.9998210169	0.9999267734	0.9999260819	0.9998348217	0.9989391598
1000	0.999740357	0.9998396556	0.9998389577	0.999747693	0.9988521022
10000	0.9989069331	0.9989444194	0.9989436602	0.9988523655	0.9979575734

Table 6: Optimal Duty Period for Software Fault Dominating Case, $R_0(\theta) = 0.9980540436$

n_1,n_2	1	6	12	1000	10000
1	0.9985237217	0.9987747407	0.9987771994	0.9986808206	0.9978013668
6	0.9999933354	0.9999949708	0.9999944509	0.9998956751	0.9989962019
12	0.9999961675	0.9999978256	0.999997306	0.9998985286	0.998999041
1000	0.9998978429	0.9998993636	0.999898842	0.9998000739	0.9989006738
10000	0.9989996472	0.9989998974	0.9989993582	0.9989006752	0.9980020843

5 Conclusion and Future Work

We have accomplished some in-depth analysis of on-board preventive maintenance for long-life deep-space mission. Our model-based evaluation not only confirms the effectiveness of the preventive maintenance strategy but also provides to us further insights regarding the tradeoffs among system/environment attributes and their collective effect on mission reliability gain from preventive maintenance. From solution method perspective, we have exploited Weibull distribution for characterizing and differentiating hardware and software aging and age-reversal processes in a natural, synergistic manner. The recurrence function described in this paper can be utilized for phased-mission analysis for a wide variety of space applications.

The analytic models presented in this paper can be further extended for the evaluation of more sophisticated on-board maintenance schemes. In particular, we plan to investigate the schemes that further utilize the X2000 architecture's scalability. Those schemes will consider the system configurations with various numbers of processor strings. For each configuration, a subset of processor string will undergo on-board preventive maintenance periodically during the mission phases that permit degradable performance. Moreover, the individual strings will be allowed to have various modes of rejuvenation, for example, operating in different reduced power levels and having different frequencies for software reinitialization.

References

[1] L. Alkalai, "NASA Center for Integrated Space Microsystems," in *Proceedings of Advanced Deep Space System Development Program Workshop on Advanced Spacecraft Technologies*, (Pasadena, CA), June 1997.

[2] Y. Huang, C. Kintala, N. Kolettis, and N. D. Fulton, "Software rejuvenation: Analysis, module and applications," in *Digest of the 25th Annual International Symposium on Fault-Tolerant Computing*, (Pasadena, CA), pp. 381–390, June 1995.

[3] S. Garg, A. Puliafito, M. Telek, and K. S. Trivedi, "Analysis of software rejuvenation using Markov regenerative stochastic Petri net," in *Proc. 6th Int'l Symposium on Software Reliability Engineering*, (Toulouse, France), pp. 180–187, Oct. 1995.

[4] P. S. Ho and T. Kwok, "Electromigration in metals," *Reports on Progress in Physics*, vol. 52, pp. 301–348, Jan. 1989.

[5] K.-N. Tu, J. W. Mayer, and L. C. Feldman, *Electronic Thin Film Science for Eletrical Engineers and Materials Scientists.* Maxwell Macmillan International, 1992.

[6] A. T. Tai, S. Chau, L. Alkalai, and H. Hecht, "On-board preventive maintenance: Analysis of effectiveness and optimal duty period," in *Proceedings of the Third International*

Workshop on Object-oriented Real-time Dependable Systems (WORDS'97), (Newport Beach, CA), pp. 40–47, Feb. 1997.

[7] L. Alkalai and M. Underwood, "Micro-electronics systems IPDT technology roadmap," Technical Report D-13276, Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA, Dec. 1995.

[8] L. Alkalai, J. Klein, and M. Underwood, "The New Millennium Program microelectronics systems, advanced technology development," in *Proceedings of the 34th Aerospace Science Meeting and Exhibit*, (Reno, Nevada), Jan. 1996.

[9] S. N. Chau, "X2000 avionics system conceptual design document," JPL Technical Report, Jet Propulsion Laboratory, California Institute of Technology, Pasadena, CA, 1997.

[10] D. Kececioglu, *Reliability engineering handbook. Volume I.* Englewood Cliffs, NJ: Prentice-Hall PTR, 1991.

[11] S. M. Ross, *Stochastic Processes.* New York: John Wiley, 1996.